In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_excel('/content/zomato_restaurants_in_India.xlsx')
```

In [3]:
```
df.head()
```

Out[3]:

	res_id	name	establishment	url	address	city	city_id	locality
0	3400299	Bikanervala	['Quick Bites']	https://www.zomato.com/agra/bikanervala-khanda...	Kalyani Point, Near Tulsi Cinema, Bypass Road,...	Agra	34	Khandari
1	3400005	Mama Chicken Mama Franky House	['Quick Bites']	https://www.zomato.com/agra/mama-chicken-mama-...	Main Market, Sadar Bazaar, Agra Cantt, Agra	Agra	34	Agra Cantt
2	3401013	Bhagat Halwai	['Quick Bites']	https://www.zomato.com/agra/bhagat-halwai-2-sh...	62/1, Near Easy Day, West Shivaji Nagar, Goalp...	Agra	34	Shahganj
3	3400290	Bhagat Halwai	['Quick Bites']	https://www.zomato.com/agra/bhagat-halwai-civi...	Near Anjana Cinema, Nehru Nagar, Civil Lines, ...	Agra	34	Civil Lines
4	3401744	The Salt Cafe Kitchen & Bar	['Casual Dining']	https://www.zomato.com/agra/the-salt-cafe-kitc...	1C,3rd Floor, Fatehabad Road, Tajganj, Agra	Agra	34	Tajganj

5 rows × 26 columns

In [4]:
```
df.tail()
```

Out[4]:

	res_id	name	establishment	url	address	city	city_
211939	3202251	Kali Mirch Cafe And Restaurant	['Casual Dining']	https://www.zomato.com/vadodara/kali-mirch-caf...	Manu Smriti Complex, Near Navrachna School, Gl...	Vadodara	3
211940	3200996	Raju Omlet	['Quick Bites']	https://www.zomato.com/vadodara/raju-omlet-kar...	Mahalaxmi Apartment, Opposite B O B, Karoli Ba...	Vadodara	3
211941	18984164	The Grand Thakar	['Casual Dining']	https://www.zomato.com/vadodara/the-grand-thak...	3rd Floor, Shreem Shalini Mall, Opposite Conqu...	Vadodara	3
211942	3201138	Subway	['Quick Bites']	https://www.zomato.com/vadodara/subway-1-akota...	G-2, Vedant Platina, Near Cosmos, Akota, Vadodara	Vadodara	3
211943	18879846	Freshco's - The Health Cafe	['CafÃ©']	https://www.zomato.com/vadodara/freshcos-the-h...	Shop 7, Ground Floor, Opposite Natubhai Circle...	Vadodara	3

5 rows × 26 columns

In [5]:
```
df.shape
```

Out[5]:
```
(211944, 26)
```

In [6]:
```
df.describe().T
```

Out[6]:

	count	mean	std	min	25%	50%	75%
res_id	211944.0	1.349411e+07	7.883722e+06	50.0	3.301027e+06	1.869573e+07	1.881297e+07
city_id	211944.0	4.746785e+03	5.568766e+03	1.0	1.100000e+01	3.400000e+01	1.130600e+04
latitude	211944.0	2.149976e+01	2.278133e+01	0.0	1.549607e+01	2.251449e+01	2.684167e+01
longitude	211944.0	7.761528e+01	7.500104e+00	0.0	7.487796e+01	7.742597e+01	8.021932e+01
country_id	211944.0	1.000000e+00	0.000000e+00	1.0	1.000000e+00	1.000000e+00	1.000000e+00
average_cost_for_two	211944.0	5.958122e+02	6.062394e+02	0.0	2.500000e+02	4.000000e+02	7.000000e+02
price_range	211944.0	1.882535e+00	8.929891e-01	1.0	1.000000e+00	2.000000e+00	2.000000e+00
aggregate_rating	211944.0	3.395937e+00	1.283642e+00	0.0	3.300000e+00	3.800000e+00	4.100000e+00
votes	211944.0	3.780019e+02	9.253334e+02	-18.0	1.600000e+01	1.000000e+02	3.620000e+02
photo_count	211944.0	2.569712e+02	8.676689e+02	0.0	3.000000e+00	1.800000e+01	1.280000e+02
opentable_support	211896.0	0.000000e+00	0.000000e+00	0.0	0.000000e+00	0.000000e+00	0.000000e+00
delivery	211944.0	-2.559072e-01	9.641721e-01	-1.0	-1.000000e+00	-1.000000e+00	1.000000e+00
takeaway	211944.0	-1.000000e+00	0.000000e+00	-1.0	-1.000000e+00	-1.000000e+00	-1.000000e+00

In [7]:
```
df.describe(include = ['object'])
```

Out[7]:

	name	establishment	url	address	city	locality	zipcode	locality
count	211944	211944	211944	211810	211944	211944	48757	
unique	41100	27	55568	50657	99	3731	1304	
top	Domino's Pizza	['Quick Bites']	https://www.zomato.com/chennai/3bs-buddies-bar...	Laxman Jhula, Tapovan, Rishikesh	Chennai	Civil Lines	0	Ana Sa
freq	3108	64390	169	299	11630	3660	9857	

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
res_id                     0
name                       0
establishment              0
url                        0
address                  134
city                       0
city_id                    0
locality                   0
latitude                   0
longitude                  0
zipcode                163187
country_id                 0
locality_verbose           0
cuisines                1391
timings                 3874
average_cost_for_two       0
price_range                0
currency                   0
highlights                 0
aggregate_rating           0
rating_text                0
votes                      0
photo_count                0
opentable_support         48
delivery                   0
takeaway                   0
dtype: int64
```

In [9]:
```
df.columns.str.strip
```

Out[9]:
pandas.core.strings.accessor.StringMethods.strip
def strip(to_strip=None)

Remove leading and trailing characters.

Strip whitespaces (including newlines) or a set of specified characters
from each string in the Series/Index from left and right sides.
Replaces any non-strings in Series with NaNs.
Equivalent to :meth:`str.strip`.

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 211944 entries, 0 to 211943
Data columns (total 26 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   res_id                211944 non-null  int64  
 1   name                  211944 non-null  object 
 2   establishment         211944 non-null  object 
 3   url                   211944 non-null  object 
 4   address               211810 non-null  object 
 5   city                  211944 non-null  object 
 6   city_id               211944 non-null  int64  
 7   locality              211944 non-null  object 
 8   latitude              211944 non-null  float64
 9   longitude             211944 non-null  float64
 10  zipcode               48757 non-null   object 
 11  country_id            211944 non-null  int64  
 12  locality_verbose      211944 non-null  object 
 13  cuisines              210553 non-null  object 
 14  timings               208070 non-null  object 
 15  average_cost_for_two  211944 non-null  int64  
 16  price_range           211944 non-null  int64  
 17  currency              211944 non-null  object 
 18  highlights            211944 non-null  object 
 19  aggregate_rating      211944 non-null  float64
 20  rating_text           211944 non-null  object 
 21  votes                 211944 non-null  int64  
 22  photo_count           211944 non-null  int64  
 23  opentable_support     211896 non-null  float64
 24  delivery              211944 non-null  int64  
 25  takeaway              211944 non-null  int64  
dtypes: float64(4), int64(9), object(13)
memory usage: 42.0+ MB
```

In [11]:
```
df.columns
```

Out[11]:
```
Index(['res_id', 'name', 'establishment', 'url', 'address', 'city', 'city_id',
       'locality', 'latitude', 'longitude', 'zipcode', 'country_id',
       'locality_verbose', 'cuisines', 'timings', 'average_cost_for_two',
       'price_range', 'currency', 'highlights', 'aggregate_rating',
       'rating_text', 'votes', 'photo_count', 'opentable_support', 'delivery',
       'takeaway'],
      dtype='object')
```

In [12]:
```
df[['address']]
```

Out[12]:

	address
0	Kalyani Point, Near Tulsi Cinema, Bypass Road,...
1	Main Market, Sadar Bazaar, Agra Cantt, Agra
2	62/1, Near Easy Day, West Shivaji Nagar, Goalp...
3	Near Anjana Cinema, Nehru Nagar, Civil Lines, ...
4	1C,3rd Floor, Fatehabad Road, Tajganj, Agra
...	...
211939	Manu Smriti Complex, Near Navrachna School, Gl...
211940	Mahalaxmi Apartment, Opposite B O B, Karoli Ba...
211941	3rd Floor, Shreem Shalini Mall, Opposite Conqu...
211942	G-2, Vedant Platina, Near Cosmos, Akota, Vadodara
211943	Shop 7, Ground Floor, Opposite Natubhai Circle...

211944 rows × 1 columns

Data Cleaning and Preparation:

In [13]:
```
for column in df:
  percentage = ((df[column].isnull().sum()/len(df))*100).round(2)
  print(f'The {column} has {percentage} percent values')
```

```
The res_id has 0.0 percent values
The name has 0.0 percent values
The establishment has 0.0 percent values
The url has 0.0 percent values
The address has 0.06 percent values
The city has 0.0 percent values
The city_id has 0.0 percent values
The locality has 0.0 percent values
The latitude has 0.0 percent values
The longitude has 0.0 percent values
The zipcode has 77.0 percent values
The country_id has 0.0 percent values
The locality_verbose has 0.0 percent values
The cuisines has 0.66 percent values
The timings has 1.83 percent values
The average_cost_for_two has 0.0 percent values
The price_range has 0.0 percent values
The currency has 0.0 percent values
The highlights has 0.0 percent values
The aggregate_rating has 0.0 percent values
The rating_text has 0.0 percent values
The votes has 0.0 percent values
The photo_count has 0.0 percent values
The opentable_support has 0.02 percent values
The delivery has 0.0 percent values
The takeaway has 0.0 percent values
```

In [14]:
```
df[['zipcode', 'cuisines']]
```

Out[14]:

	zipcode	cuisines
0	NaN	North Indian, South Indian, Mithai, Street Foo...
1	282001	North Indian, Mughlai, Rolls, Chinese, Fast Fo...
2	282010	Fast Food, Mithai
3	282002	Desserts, Bakery, Fast Food, South Indian
4	NaN	North Indian, Continental, Italian
...	...	...
211939	390024	North Indian
211940	NaN	Fast Food
211941	NaN	Gujarati, North Indian, Chinese
211942	NaN	Fast Food, Sandwich, Salad
211943	390007	Cafe, Healthy Food, Coffee

211944 rows × 2 columns

In [15]:
```
df.drop(['zipcode'],axis=1,inplace=True)
```

In [16]:
```
df.drop(['url'],axis=1,inplace=True)
df.drop(['address'],axis=1,inplace=True)
df.drop(['country_id'],axis=1,inplace=True)
df.drop(['timings'],axis=1,inplace=True)
df.drop(['currency'],axis=1,inplace=True)
df.drop(['opentable_support'],axis=1,inplace=True)
```

In [17]:
```
df.columns
```

Out[17]:
```
Index(['res_id', 'name', 'establishment', 'city', 'city_id', 'locality',
       'latitude', 'longitude', 'locality_verbose', 'cuisines',
       'average_cost_for_two', 'price_range', 'highlights', 'aggregate_rating',
       'rating_text', 'votes', 'photo_count', 'delivery', 'takeaway'],
      dtype='object')
```

In [18]:
```
df['cuisines'].fillna('unknown', inplace = True)
```

In [19]:
```
df.isnull().sum()
```

Out[19]:
```
res_id                  0
name                    0
establishment           0
city                    0
city_id                 0
locality                0
latitude                0
longitude               0
locality_verbose        0
cuisines                0
average_cost_for_two    0
price_range             0
highlights              0
aggregate_rating        0
rating_text             0
votes                   0
photo_count             0
delivery                0
takeaway                0
dtype: int64
```

In [20]:
```
df.duplicated().sum()
```

Out[20]:
```
151533
```

In [21]:
```
duplicate = ((df.duplicated().sum()/len(df))*100).round(2)
print(f'The data has {duplicate} percent duplicate values')
```

```
The data has 71.5 percent duplicate values
```

In [22]:
```
df2 = df.drop_duplicates(keep = 'first')
```

In [23]:
```
df2
```

Out[23]:

	res_id	name	establishment	city	city_id	locality	latitude	longitude	locality_verbose
0	3400299	Bikanervala	['Quick Bites']	Agra	34	Khandari	27.211450	78.002381	Khandari, Agra
1	3400005	Mama Chicken Mama Franky House	['Quick Bites']	Agra	34	Agra Cantt	27.160569	78.011583	Agra Cantt, Agra
2	3401013	Bhagat Halwai	['Quick Bites']	Agra	34	Shahganj	27.182938	77.979684	Shahganj, Agra
3	3400290	Bhagat Halwai	['Quick Bites']	Agra	34	Civil Lines	27.205668	78.004799	Civil Lines, Agra
4	3401744	The Salt Cafe Kitchen & Bar	['Casual Dining']	Agra	34	Tajganj	27.157709	78.052421	Tajganj, Agra
...	...	...	...	...	...	...	...	...	...
211882	19142822	Shree Janta Ice Cream	['Dessert Parlour']	Vadodara	32	Manjalpur	22.270516	73.196408	Manjalpur, Vadodara
211925	18984164	The Grand Thakar	['Casual Dining']	Vadodara	32	Alkapuri	22.310563	73.171163	Alkapuri, Vadodara
211926	18019952	Geeta lodge	['Casual Dining']	Vadodara	32	Alkapuri	22.317731	73.168107	Alkapuri, Vadodara
211940	3200996	Raju Omlet	['Quick Bites']	Vadodara	32	Karelibaug	22.322455	73.197203	Karelibaug, Vadodara
211942	3201138	Subway	['Quick Bites']	Vadodara	32	Akota	22.270027	73.143068	Akota, Vadodara

	res_id	name	establishment	city	city_id	locality	latitude	longitude	locality_verbose

60411 rows × 19 columns

In [24]:
```
df2.shape
```

Out[24]:
```
(60411, 19)
```

In [25]:
```
df2.duplicated().sum()
```

Out[25]:
```
0
```

In [26]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 60411 entries, 0 to 211942
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   res_id                60411 non-null  int64  
 1   name                  60411 non-null  object 
 2   establishment         60411 non-null  object 
 3   city                  60411 non-null  object 
 4   city_id               60411 non-null  int64  
 5   locality              60411 non-null  object 
 6   latitude              60411 non-null  float64
 7   longitude             60411 non-null  float64
 8   locality_verbose      60411 non-null  object 
 9   cuisines              60411 non-null  object 
 10  average_cost_for_two  60411 non-null  int64  
 11  price_range           60411 non-null  int64  
 12  highlights            60411 non-null  object 
 13  aggregate_rating      60411 non-null  float64
 14  rating_text           60411 non-null  object 
 15  votes                 60411 non-null  int64  
 16  photo_count           60411 non-null  int64  
 17  delivery              60411 non-null  int64  
 18  takeaway              60411 non-null  int64  
dtypes: float64(3), int64(8), object(8)
memory usage: 9.2+ MB
```

In [27]:
```
df2['res_id'].duplicated()
```

Out[27]:
```
0         False
1         False
2         False
3         False
4         False
          ...
211882    False
211925    False
211926    False
211940    False
211942    False
Name: res_id, Length: 60411, dtype: bool
```

In [28]:
```
df2['establishment'].value_counts()
```

Out[28]:
```
establishment
['Quick Bites']        15473
['Casual Dining']      13761
['CafÃ©']               4644
['Dessert Parlour']     3915
['Bakery']              3887
['Sweet Shop']          2712
['Beverage Shop']       2566
[]                      1920
['Fine Dining']         1656
['Food Court']          1569
['Bar']                 1550
['Dhaba']               1334
['Kiosk']               1196
['Lounge']               898
['Food Truck']           874
['Bhojanalya']           654
['Mess']                 397
['Pub']                  393
['Paan Shop']            326
['Confectionery']        227
['Butcher Shop']         154
['Microbrewery']         136
['Club']                 113
['Shack']                 21
['Cocktail Bar']          17
['Irani Cafe']            14
['Pop up']                 4
Name: count, dtype: int64
```

In [29]:
```
df2['establishment'] = df2['establishment'].str.replace('@', '').str.replace('cafA', 'cafe')
print(df2)
```

```
          res_id                            name        establishment  \
0        3400299                     Bikanervala      ['Quick Bites']
1        3400005  Mama Chicken Mama Franky House      ['Quick Bites']
2        3401013                  Bhagat Halwai      ['Quick Bites']
3        3400290                  Bhagat Halwai      ['Quick Bites']
4        3401744     The Salt Cafe Kitchen & Bar    ['Casual Dining']
...          ...                             ...                  ...
211882  19142822           Shree Janta Ice Cream  ['Dessert Parlour']
211925  18984164                The Grand Thakar    ['Casual Dining']
211926  18019952                     Geeta lodge    ['Casual Dining']
211940   3200996                      Raju Omlet      ['Quick Bites']
211942   3201138                          Subway      ['Quick Bites']

            city  city_id     locality   latitude  longitude  \
0          Agra       34     Khandari  27.211450  78.002381
1          Agra       34   Agra Cantt  27.160569  78.011583
2          Agra       34     Shahganj  27.182938  77.979684
3          Agra       34  Civil Lines  27.205668  78.004799
4          Agra       34      Tajganj  27.157709  78.052421
...         ...      ...          ...        ...        ...
211882  Vadodara       32    Manjalpur  22.270516  73.196408
211925  Vadodara       32     Alkapuri  22.310563  73.171163
211926  Vadodara       32     Alkapuri  22.317731  73.168107
211940  Vadodara       32   Karelibaug  22.322455  73.197203
211942  Vadodara       32        Akota  22.270027  73.143068

            locality_verbose  \
0             Khandari, Agra
1          Agra Cantt, Agra
2             Shahganj, Agra
3         Civil Lines, Agra
4              Tajganj, Agra
...                      ...
211882   Manjalpur, Vadodara
211925    Alkapuri, Vadodara
211926    Alkapuri, Vadodara
211940  Karelibaug, Vadodara
211942       Akota, Vadodara

                                              cuisines  \
0       North Indian, South Indian, Mithai, Street Foo...
1       North Indian, Mughlai, Rolls, Chinese, Fast Fo...
2                                    Fast Food, Mithai
3              Desserts, Bakery, Fast Food, South Indian
4                     North Indian, Continental, Italian
...                                                 ...
211882                                        Ice Cream
211925                   Gujarati, North Indian, Chinese
211926                             Gujarati, Street Food
211940                                         Fast Food
211942                       Fast Food, Sandwich, Salad

        average_cost_for_two  price_range  \
0                        700            2
1                        600            2
2                        300            1
3                        300            1
4                       1000            3
...                      ...          ...
211882                   200            1
211925                   700            2
211926                   250            1
```

```
211940                  300           1
211942                  500           2

                                                highlights  aggregate_rating  \
0       ['Lunch', 'Takeaway Available', 'Credit Card',...               4.4
1       ['Delivery', 'No Alcohol Available', 'Dinner',...               4.4
2       ['No Alcohol Available', 'Dinner', 'Takeaway A...               4.2
3       ['Takeaway Available', 'Credit Card', 'Lunch',...               4.3
4       ['Lunch', 'Serves Alcohol', 'Cash', 'Credit Ca...               4.9
...                                                   ...               ...
211882  ['Cash', 'Takeaway Available', 'Delivery', 'In...               2.9
211925  ['Dinner', 'Cash', 'Debit Card', 'Lunch', 'Tak...               4.0
211926  ['Dinner', 'Cash', 'Credit Card', 'Lunch', 'Ta...               3.9
211940  ['Dinner', 'Cash', 'Takeaway Available', 'Debi...               4.1
211942  ['Dinner', 'Delivery', 'Credit Card', 'Lunch',...               3.7

       rating_text  votes  photo_count  delivery  takeaway
0        Very Good    814          154        -1        -1
1        Very Good   1203          161        -1        -1
2        Very Good    801          107         1        -1
3        Very Good    693          157         1        -1
4        Excellent    470          291         1        -1
...            ...    ...          ...       ...       ...
211882     Average      4            1         1        -1
211925   Very Good    111           38        -1        -1
211926        Good    207           14        -1        -1
211940   Very Good    187           40         1        -1
211942        Good    128           34         1        -1

[60411 rows x 19 columns]
<ipython-input-29-5969885dec1e>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  df2['establishment'] = df2['establishment'].str.replace('@', '').str.replace('cafA', 'cafe')
```

In [30]:
```
df2['establishment'] = df2['establishment'].str.replace(r'\[]', 'Not Specified', regex=True)
print(df2)
```

```
          res_id                            name        establishment  \
0        3400299                     Bikanervala      ['Quick Bites']
1        3400005  Mama Chicken Mama Franky House      ['Quick Bites']
2        3401013                  Bhagat Halwai      ['Quick Bites']
3        3400290                  Bhagat Halwai      ['Quick Bites']
4        3401744     The Salt Cafe Kitchen & Bar    ['Casual Dining']
...          ...                             ...                  ...
211882  19142822           Shree Janta Ice Cream  ['Dessert Parlour']
211925  18984164                The Grand Thakar    ['Casual Dining']
211926  18019952                     Geeta lodge    ['Casual Dining']
211940   3200996                      Raju Omlet      ['Quick Bites']
211942   3201138                          Subway      ['Quick Bites']

            city  city_id     locality   latitude  longitude  \
0          Agra       34     Khandari  27.211450  78.002381
1          Agra       34   Agra Cantt  27.160569  78.011583
2          Agra       34     Shahganj  27.182938  77.979684
3          Agra       34  Civil Lines  27.205668  78.004799
4          Agra       34      Tajganj  27.157709  78.052421
...         ...      ...          ...        ...        ...
211882  Vadodara       32    Manjalpur  22.270516  73.196408
211925  Vadodara       32     Alkapuri  22.310563  73.171163
211926  Vadodara       32     Alkapuri  22.317731  73.168107
211940  Vadodara       32   Karelibaug  22.322455  73.197203
211942  Vadodara       32        Akota  22.270027  73.143068

            locality_verbose  \
0             Khandari, Agra
1          Agra Cantt, Agra
2             Shahganj, Agra
3         Civil Lines, Agra
4              Tajganj, Agra
...                      ...
211882   Manjalpur, Vadodara
211925    Alkapuri, Vadodara
211926    Alkapuri, Vadodara
211940  Karelibaug, Vadodara
211942       Akota, Vadodara

                                              cuisines  \
0       North Indian, South Indian, Mithai, Street Foo...
1       North Indian, Mughlai, Rolls, Chinese, Fast Fo...
2                                    Fast Food, Mithai
3              Desserts, Bakery, Fast Food, South Indian
4                     North Indian, Continental, Italian
...                                                 ...
211882                                        Ice Cream
211925                   Gujarati, North Indian, Chinese
211926                             Gujarati, Street Food
211940                                         Fast Food
211942                       Fast Food, Sandwich, Salad

        average_cost_for_two  price_range  \
0                        700            2
1                        600            2
2                        300            1
3                        300            1
4                       1000            3
...                      ...          ...
211882                   200            1
211925                   700            2
211926                   250            1
```

```
211940                  300           1
211942                  500           2

                                                highlights  aggregate_rating  \
0       ['Lunch', 'Takeaway Available', 'Credit Card',...               4.4
1       ['Delivery', 'No Alcohol Available', 'Dinner',...               4.4
2       ['No Alcohol Available', 'Dinner', 'Takeaway A...               4.2
3       ['Takeaway Available', 'Credit Card', 'Lunch',...               4.3
4       ['Lunch', 'Serves Alcohol', 'Cash', 'Credit Ca...               4.9
...                                                   ...               ...
211882  ['Cash', 'Takeaway Available', 'Delivery', 'In...               2.9
211925  ['Dinner', 'Cash', 'Debit Card', 'Lunch', 'Tak...               4.0
211926  ['Dinner', 'Cash', 'Credit Card', 'Lunch', 'Ta...               3.9
211940  ['Dinner', 'Cash', 'Takeaway Available', 'Debi...               4.1
211942  ['Dinner', 'Delivery', 'Credit Card', 'Lunch',...               3.7

       rating_text  votes  photo_count  delivery  takeaway
0        Very Good    814          154        -1        -1
1        Very Good   1203          161        -1        -1
2        Very Good    801          107         1        -1
3        Very Good    693          157         1        -1
4        Excellent    470          291         1        -1
...            ...    ...          ...       ...       ...
211882     Average      4            1         1        -1
211925   Very Good    111           38        -1        -1
211926        Good    207           14        -1        -1
211940   Very Good    187           40         1        -1
211942        Good    128           34         1        -1

[60411 rows x 19 columns]
<ipython-input-30-bbb73a422595>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy
  df2['establishment'] = df2['establishment'].str.replace(r'\[]', 'Not Specified', regex=True)
```

Exploratory Data Analysis:

In [31]:
```
df2.shape
```

Out[31]:
```
(60411, 19)
```

In [32]:
```
descriptive_stats = df2.describe()

print("Descriptive Statistics:\n", descriptive_stats)
```

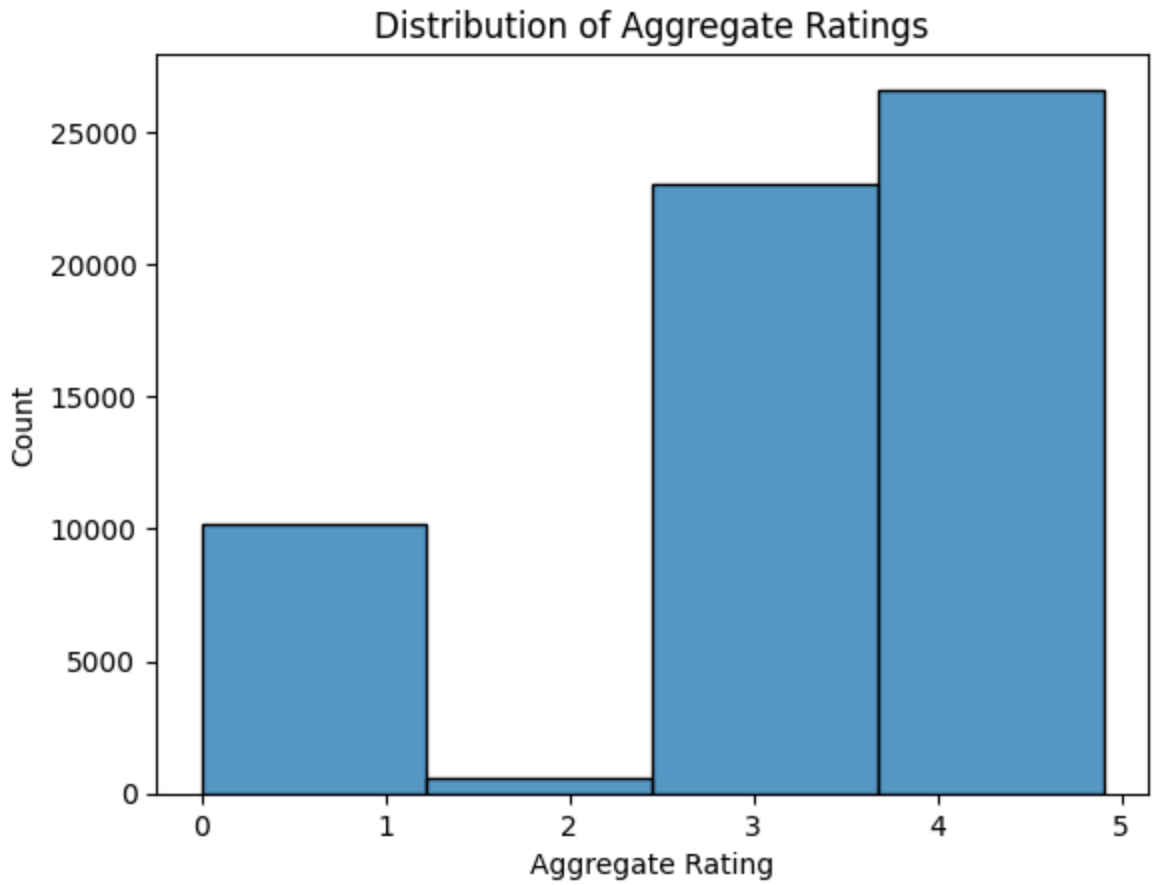
```
Descriptive Statistics:
              res_id       city_id      latitude     longitude  \
count  6.041100e+04  60411.000000  60411.000000  60411.000000
mean   1.309279e+07   3417.519376     21.349912     76.587636
std    8.133021e+06   5179.013230     41.190015     10.600963
min    5.000000e+01      1.000000      0.000000      0.000000
25%    3.000479e+06      7.000000     16.323783     74.653081
50%    1.869150e+07     26.000000     22.320915     77.134838
75%    1.886668e+07  11295.000000     26.744393     79.928133
max    1.915979e+07  11354.000000  10000.000000     91.832769

       average_cost_for_two   price_range  aggregate_rating         votes  \
count           60411.00000  60411.000000      60411.000000  60411.000000
mean              538.31246      1.730844          3.032863    261.587062
std               593.86415      0.880470          1.440739    728.316928
min                 0.00000      1.000000          0.000000    -18.000000
25%               200.00000      1.000000          2.900000      7.000000
50%               400.00000      1.000000          3.500000     42.000000
75%               600.00000      2.000000          4.000000    207.000000
max             30000.00000      4.000000          4.900000  42539.000000

        photo_count      delivery  takeaway
count  60411.000000  60411.000000   60411.0
mean     194.262303     -0.371737      -1.0
std      705.715642      0.925274       0.0
min        0.000000     -1.000000      -1.0
25%        1.000000     -1.000000      -1.0
50%       11.000000     -1.000000      -1.0
75%       82.000000      1.000000      -1.0
max    17702.000000      1.000000      -1.0
```

In [33]:
```
df2.describe(include = ['object'])
```

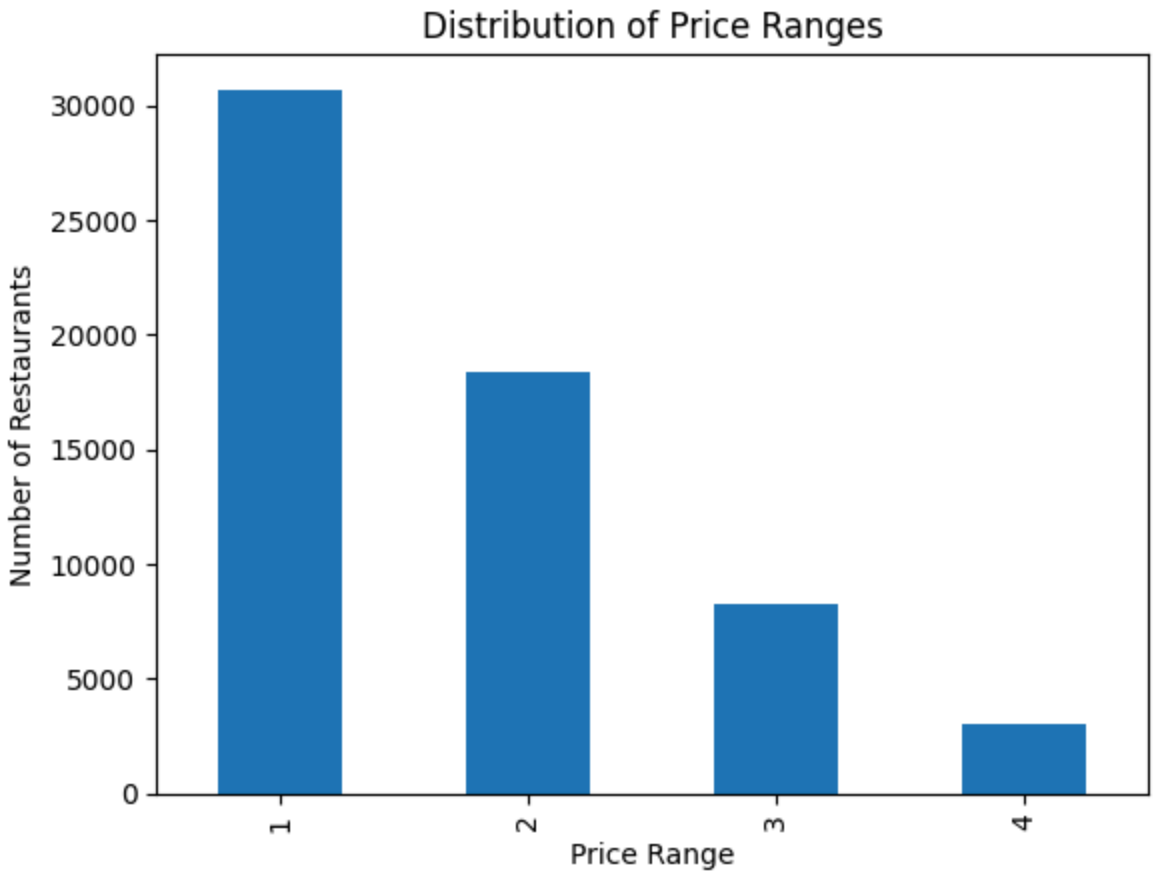
Out[33]:

	name	establishment	city	locality	locality_verbose	cuisines	highlights	rating_text
count	60411	60411	60411	60411	60411	60411	60411	60411
unique	41100	27	99	3731	3910	9383	31455	39
top	Domino's Pizza	['Quick Bites']	Chennai	Civil Lines	Gomti Nagar, Lucknow	North Indian	['Dinner', 'Takeaway Available', 'Lunch', 'Cas...	Good
freq	406	15473	2612	804	315	4587	925	17569

In [34]:
```
ratings = df2['aggregate_rating']
plot = sns.histplot(ratings, bins=4) # Using histplot for histogram
plot.set_xlabel("Aggregate Rating")  # Set x-axis label
plot.set_ylabel("Count")  # Set y-axis label
plot.set_title("Distribution of Aggregate Ratings")  # Set title
plt.show()
```



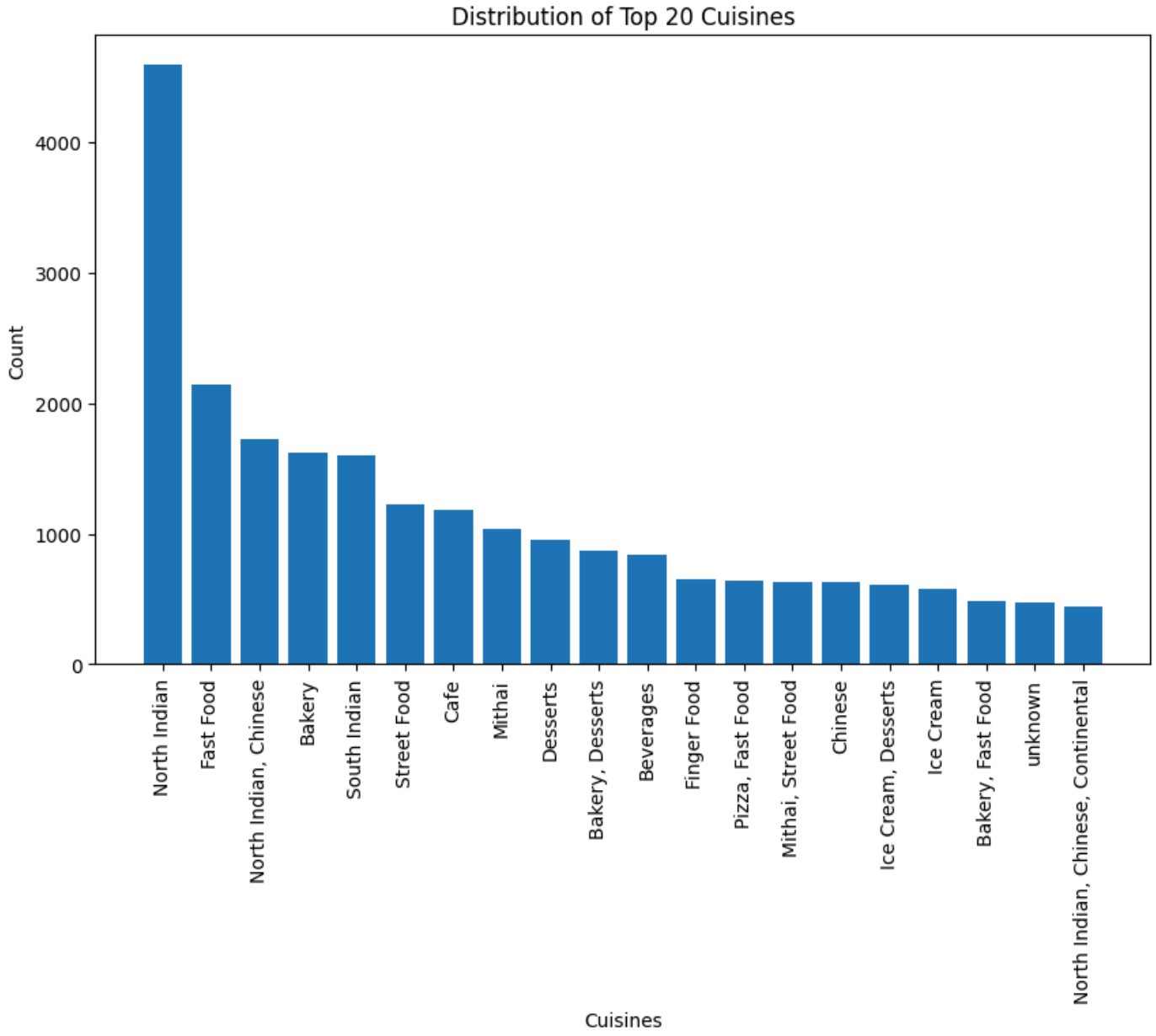
In [35]:
```
df2['price_range'].value_counts().plot(kind='bar')
plt.title('Distribution of Price Ranges')
plt.xlabel('Price Range')
plt.ylabel('Number of Restaurants')
plt.show()
```



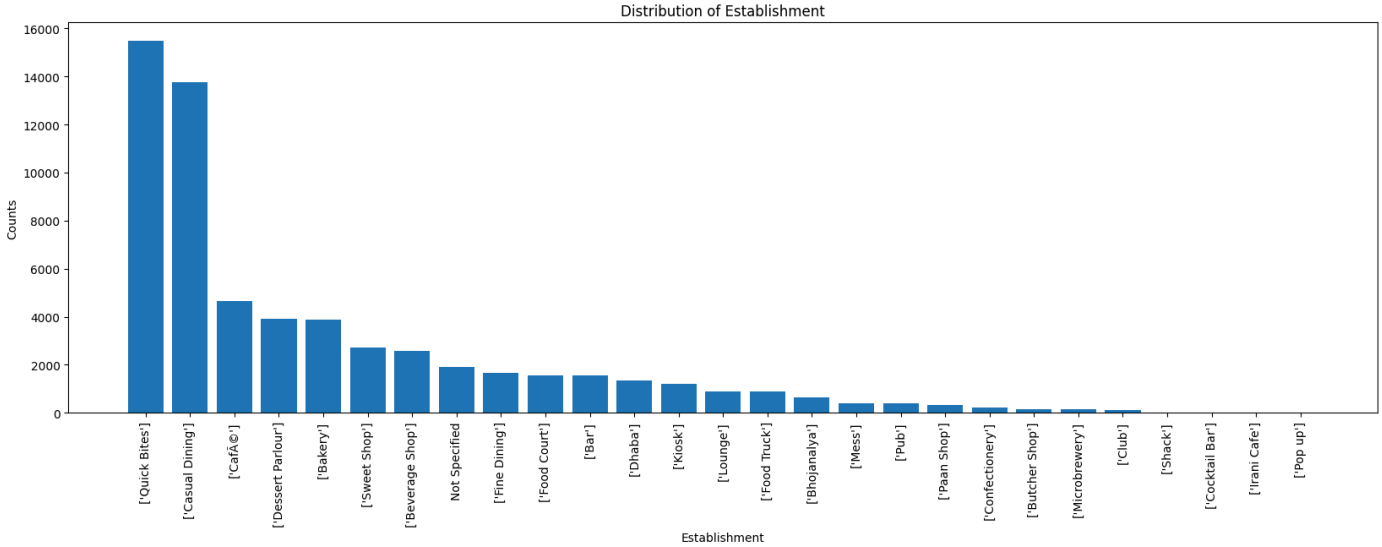
In [36]:
```
cuisines2 = df2['cuisines'].value_counts()[:20]
cuisines2
```

Out[36]:
```
cuisines
North Indian                          4587
Fast Food                             2137
North Indian, Chinese                 1720
Bakery                                1618
South Indian                          1598
Street Food                           1221
Cafe                                  1180
Mithai                                1032
Desserts                               950
Bakery, Desserts                       872
Beverages                              839
Finger Food                            649
Pizza, Fast Food                       642
Mithai, Street Food                    635
Chinese                                633
Ice Cream, Desserts                    612
Ice Cream                              576
Bakery, Fast Food                      484
unknown                                470
North Indian, Chinese, Continental     447
Name: count, dtype: int64
```

In [37]:
```
cuisines = df2['cuisines'].value_counts()[:20]
plt.figure(figsize=(10, 6))
plot = plt.bar(cuisines.index,cuisines.values)
plt.title('Distribution of Top 20 Cuisines')
plt.xticks(rotation=90)
plt.xlabel('Cuisines')
plt.ylabel('Count')
plt.show()
```



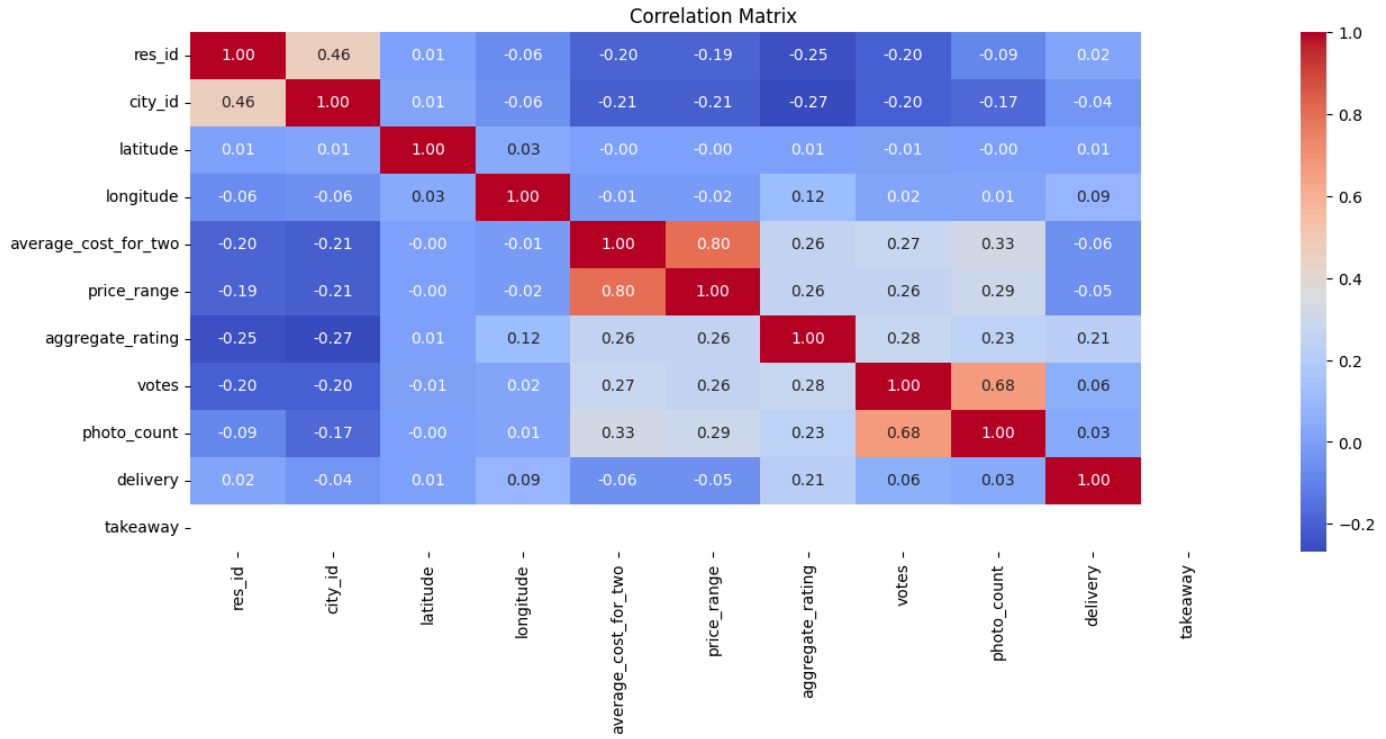
In [38]:
```
est = df2['establishment'].value_counts()
plt.figure(figsize=(20,6))
plot = plt.bar(est.index,est.values)
plt.title('Distribution of Establishment')
plt.xticks(rotation = 90)
plt.xlabel('Establishment')
plt.ylabel('Counts')
plt.show()
```



In [39]:
```
df2['establishment'].value_counts()
```

Out[39]:
```
establishment
['Quick Bites']        15473
['Casual Dining']      13761
['CafÃ©']                4644
['Dessert Parlour']     3915
['Bakery']              3887
['Sweet Shop']          2712
['Beverage Shop']       2566
Not Specified           1920
['Fine Dining']         1656
['Food Court']          1569
['Bar']                 1550
['Dhaba']               1334
['Kiosk']               1196
['Lounge']               898
['Food Truck']           874
['Bhojanalya']           654
['Mess']                 397
['Pub']                  393
['Paan Shop']            326
['Confectionery']        227
['Butcher Shop']         154
['Microbrewery']         136
['Club']                 113
['Shack']                 21
['Cocktail Bar']          17
['Irani Cafe']            14
['Pop up']                 4
Name: count, dtype: int64
```

In [40]:
```
numerical_variable = df2.select_dtypes(include=['number'])
plt.figure(figsize=(15,6))
correlation_matrix =numerical_variable.corr()
sns.heatmap(correlation_matrix, annot = True, cmap = 'coolwarm', fmt = '.2f')
plt.title('Correlation Matrix')
plt.show()
```



Regional Analysis:

In [41]:
```
df2.head(1)
```

Out[41]:

	res_id	name	establishment	city	city_id	locality	latitude	longitude	locality_verbose	cuisines	ave
0	3400299	Bikanervala	['Quick Bites']	Agra	34	Khandari	27.21145	78.002381	Khandari, Agra	North Indian, South Indian, Mithai, Street Foo...	

Top cuisines by city and ratings

In [42]:
```
region = df2.groupby(['city','cuisines'])['aggregate_rating'].mean().reset_index()
regionsorted = region.sort_values(by = ['city','aggregate_rating'], ascending = [True, False])
topcuisines = regionsorted.groupby('city').head(1)
topcuisines
```

Out[42]:

	city	cuisines	aggregate_rating
168	Agra	North Indian, Continental, Italian	4.9
304	Ahmedabad	Chinese, Japanese	4.9
587	Ajmer	Continental, Beverages, South Indian, Fast Foo...	4.8
715	Alappuzha	Arabian, Continental	4.0
921	Allahabad	North Indian, Mediterranean	4.7
...	...	...	...
20529	Varanasi	North Indian, Chinese, BBQ	4.9
20697	Vellore	Juices, Italian, Burger	3.8
20903	Vijayawada	North Indian, Andhra	4.9
21108	Vizag	European, Mediterranean, North Indian	4.9
21243	Zirakpur	Andhra, Goan, North Indian, Kerala	4.6

99 rows × 3 columns

Distribution of restaurant counts in city

In [43]:
```
estb = df2.groupby('city')['establishment'].size().reset_index(name = 'count')
estb1 = estb.sort_values(by='count', ascending=False)
estb1
```

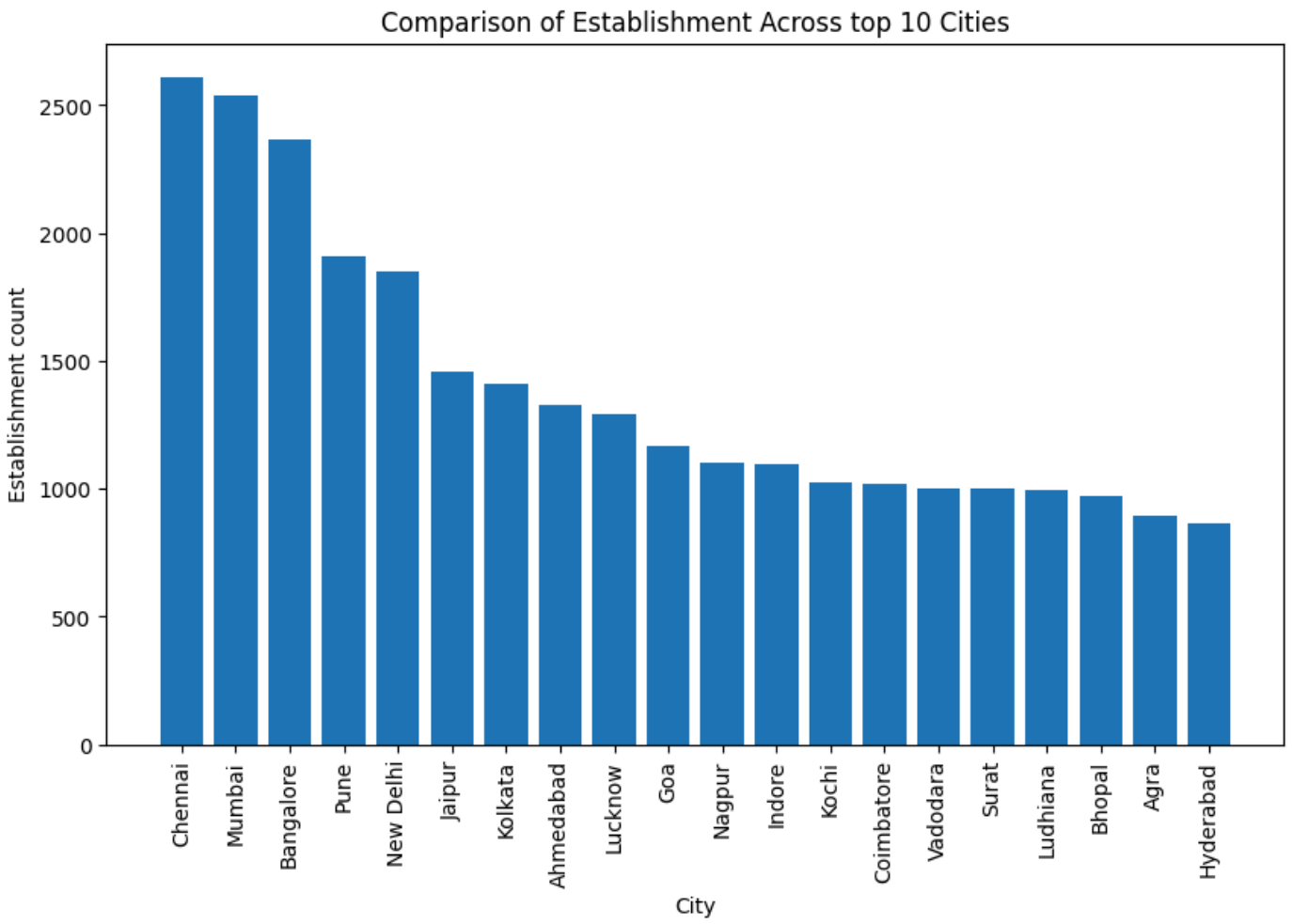
Out[43]:

	city	count
12	Chennai	2612
56	Mumbai	2538
8	Bangalore	2365
74	Pune	1911
66	New Delhi	1847
...	...	...
92	Udupi	61
30	Howrah	50
65	Neemrana	26
24	Greater Noida	22
64	Nayagaon	15

99 rows × 2 columns

In [44]:
```
top10estb = estb1.head(20)
plt.figure(figsize=(10, 6))
plt.bar(top10estb['city'], top10estb['count'])
# Add labels and title
plt.xlabel('City')
```

```
plt.ylabel('Establishment count')
plt.title('Comparison of Establishment Across top 10 Cities')
plt.xticks(rotation=90)

# Show the plot
plt.show()
```



Price Range across cities

In [45]:
```
df2[['price_range']]
```

Out[45]:

	price_range
0	2
1	2
2	1
3	1
4	3
...	...
211882	1
211925	2
211926	1
211940	1
211942	2

60411 rows × 1 columns

In [46]:
```
avgprice = df2.groupby('city')['price_range'].mean().reset_index(name = 'Average Prices')
avgpricesort = avgprice.sort_values(by = 'Average Prices', ascending = False)
avgpricesort
```

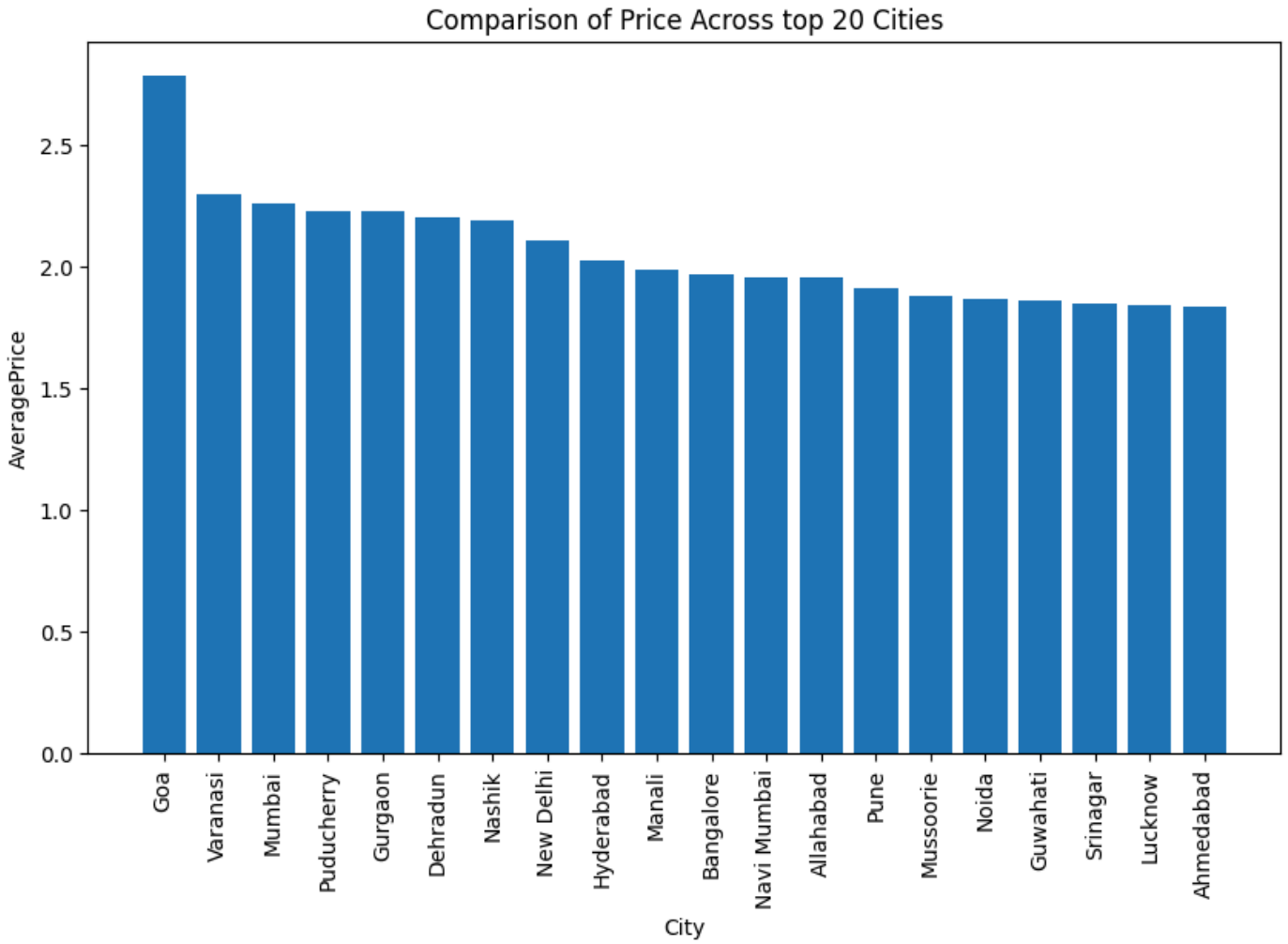
Out[46]:

	city	Average Prices
22	Goa	2.786997
94	Varanasi	2.299331
56	Mumbai	2.257289
73	Puducherry	2.226415
26	Gurgaon	2.226244
...	...	...
64	Nayagaon	1.266667
43	Kharagpur	1.181034
37	Jamnagar	1.110588
5	Amravati	1.097727
41	Junagadh	1.060606

99 rows × 2 columns

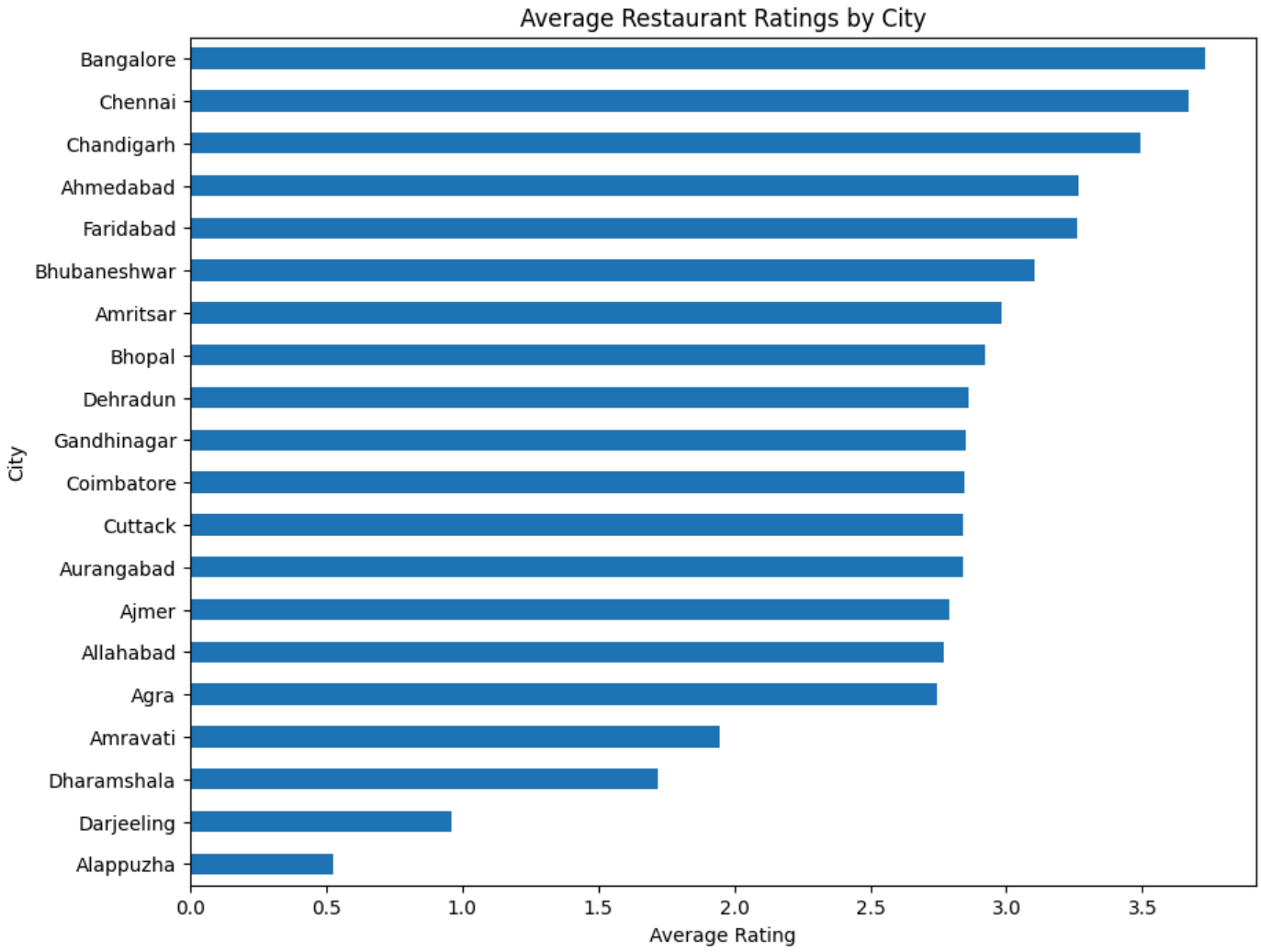
In [47]:
```
top20price = avgpricesort.head(20)
plt.figure(figsize=(10, 6))
plt.bar(top20price['city'], top20price['Average Prices'])
# Add labels and title
plt.xlabel('City')
plt.ylabel('AveragePrice')
plt.title('Comparison of Price Across top 20 Cities')
plt.xticks(rotation=90)
```

```
# Show the plot
plt.show()
```



Restaurant ratings by City

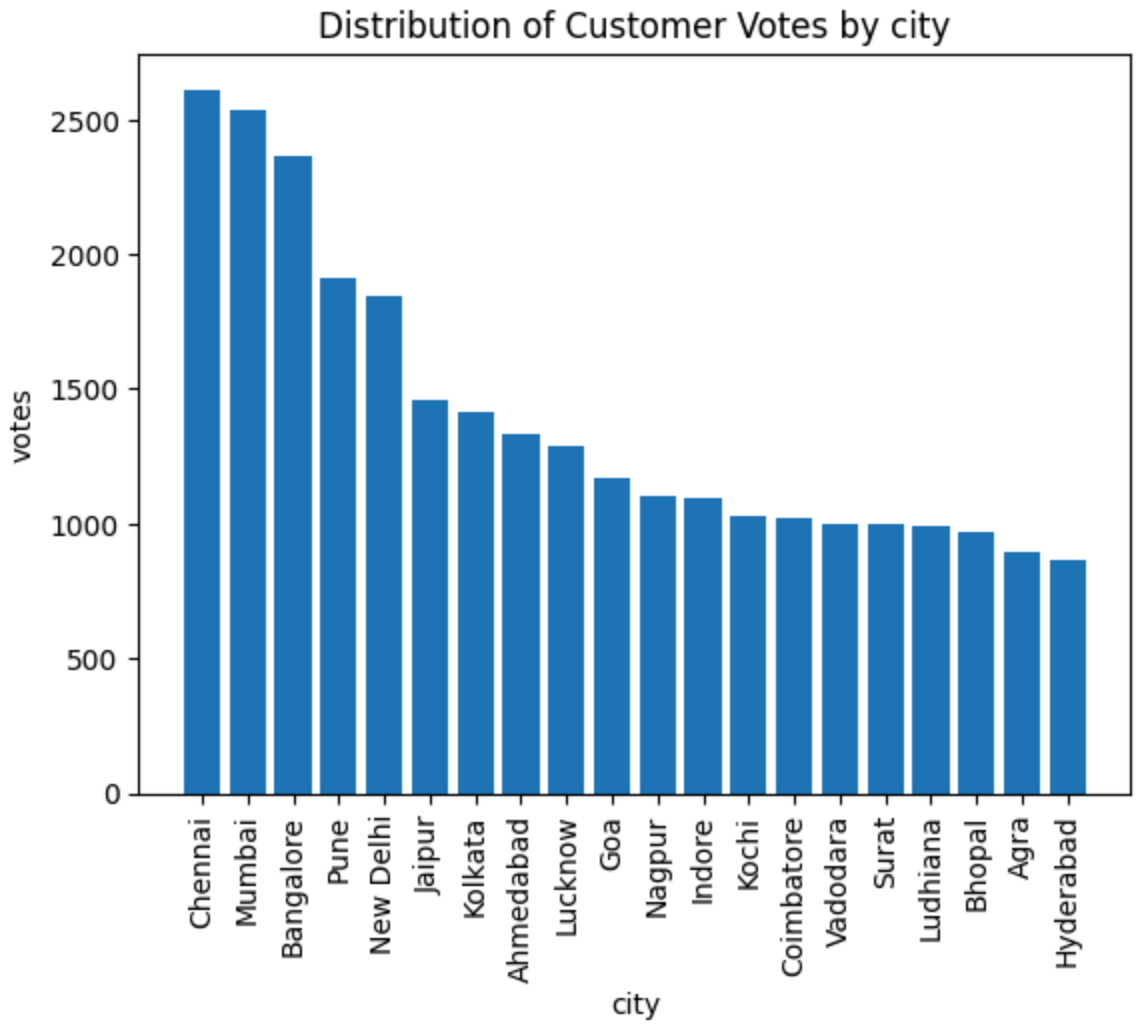
In [48]:
```
average_ratings = df2.groupby('city')['aggregate_rating'].mean()
avg = average_ratings.head(20)
# Plotting
avg.sort_values().plot(kind='barh', figsize=(10, 8))

plt.title('Average Restaurant Ratings by City')
plt.xlabel('Average Rating')
plt.ylabel('City')
plt.show()
```



Customer Votes by City

In [49]:

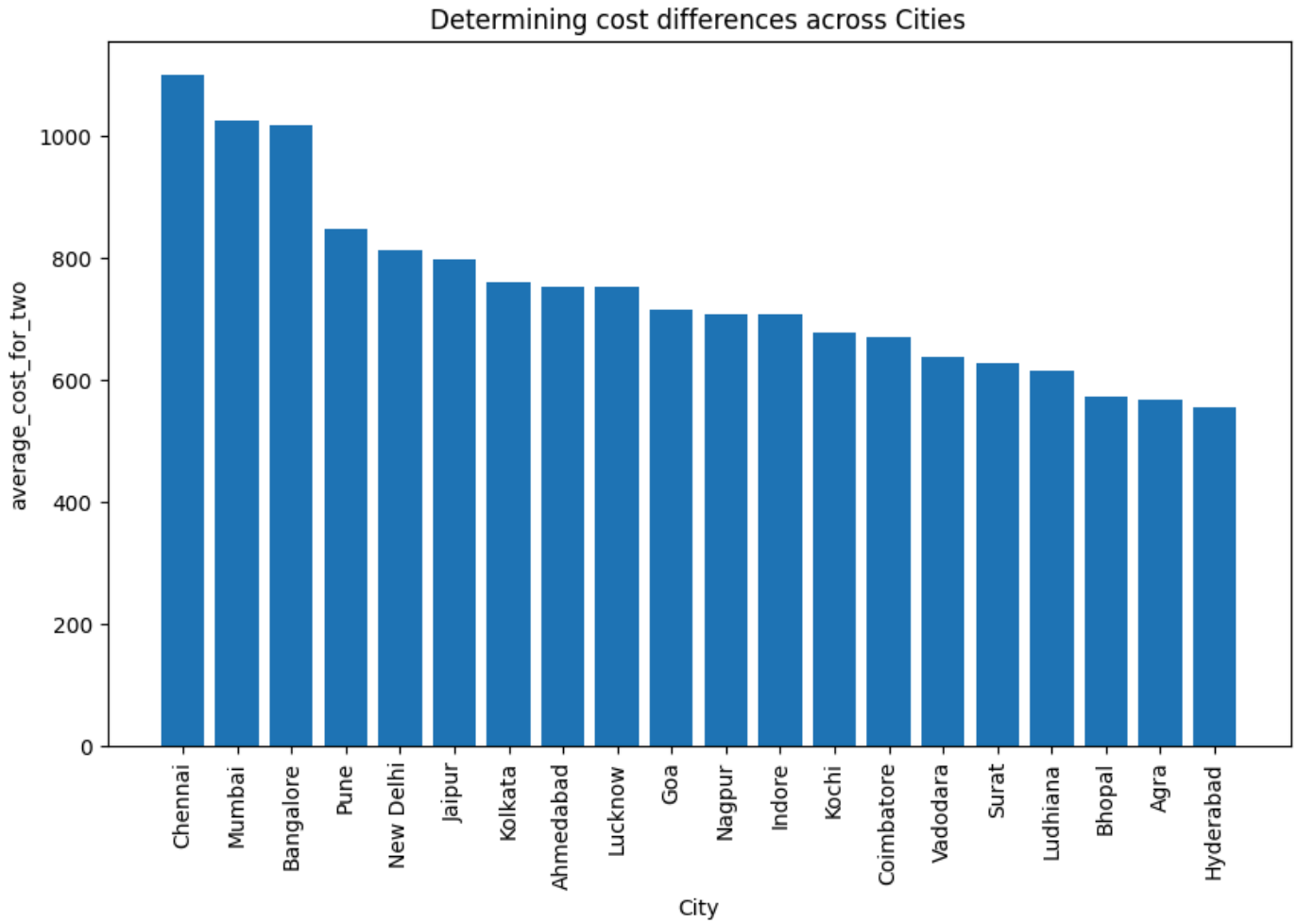
```
voting = df2.groupby('city')['votes'].size().reset_index()
vot = voting.sort_values(by = 'votes', ascending = False)
vot = vot.head(20)
plt.bar(vot['city'],vot['votes'])
plt.title('Distribution of Customer Votes by city')
plt.xlabel('city')
plt.ylabel('votes')
plt.xticks(rotation = 90)
plt.show()
```



Cost Diferences by City

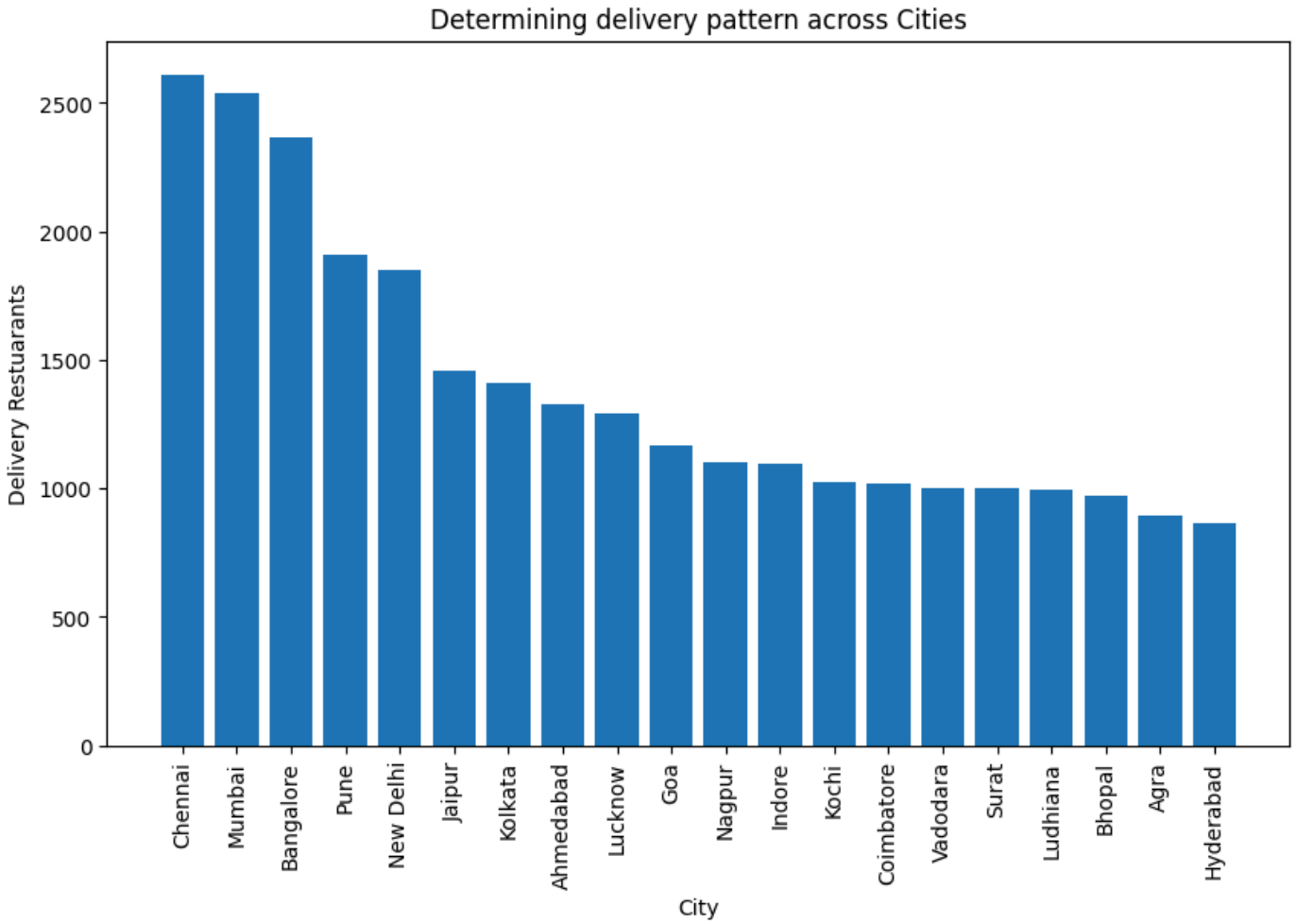
In [50]:

```
cost = df2.groupby('city')['average_cost_for_two'].mean().reset_index()
cost1 = cost.sort_values(by='average_cost_for_two', ascending=False)

# Select the top 20 rows
top20cost = cost1.head(20)


plt.figure(figsize=(10, 6))
plt.bar(top10estb['city'], top20cost['average_cost_for_two'])
# Add labels and title
plt.xlabel('City')
plt.ylabel('average_cost_for_two')
plt.title('Determining cost differences across Cities')
plt.xticks(rotation=90)

# Show the plot
plt.show()
```



In [51]:

```
estb = df2.groupby('city')['delivery'].size().reset_index()
estb1 = estb.sort_values(by='delivery', ascending=False)

# Select the top 20 rows
top10estb = estb1.head(20)


plt.figure(figsize=(10, 6))
plt.bar(top10estb['city'], top10estb['delivery'])
# Add labels and title
plt.xlabel('City')
plt.ylabel('Delivery Restuarants')
plt.title('Determining delivery pattern across Cities')
plt.xticks(rotation=90)

# Show the plot
plt.show()
```



Customer Preference Analysis:

In [52]:
```
cuisi = df2.groupby(['city','cuisines'])['aggregate_rating'].mean().reset_index()
cussort = cuisi.sort_values(by = ['city','aggregate_rating'], ascending =  False)
cus = cussort.groupby('city').head(1)
cus
```

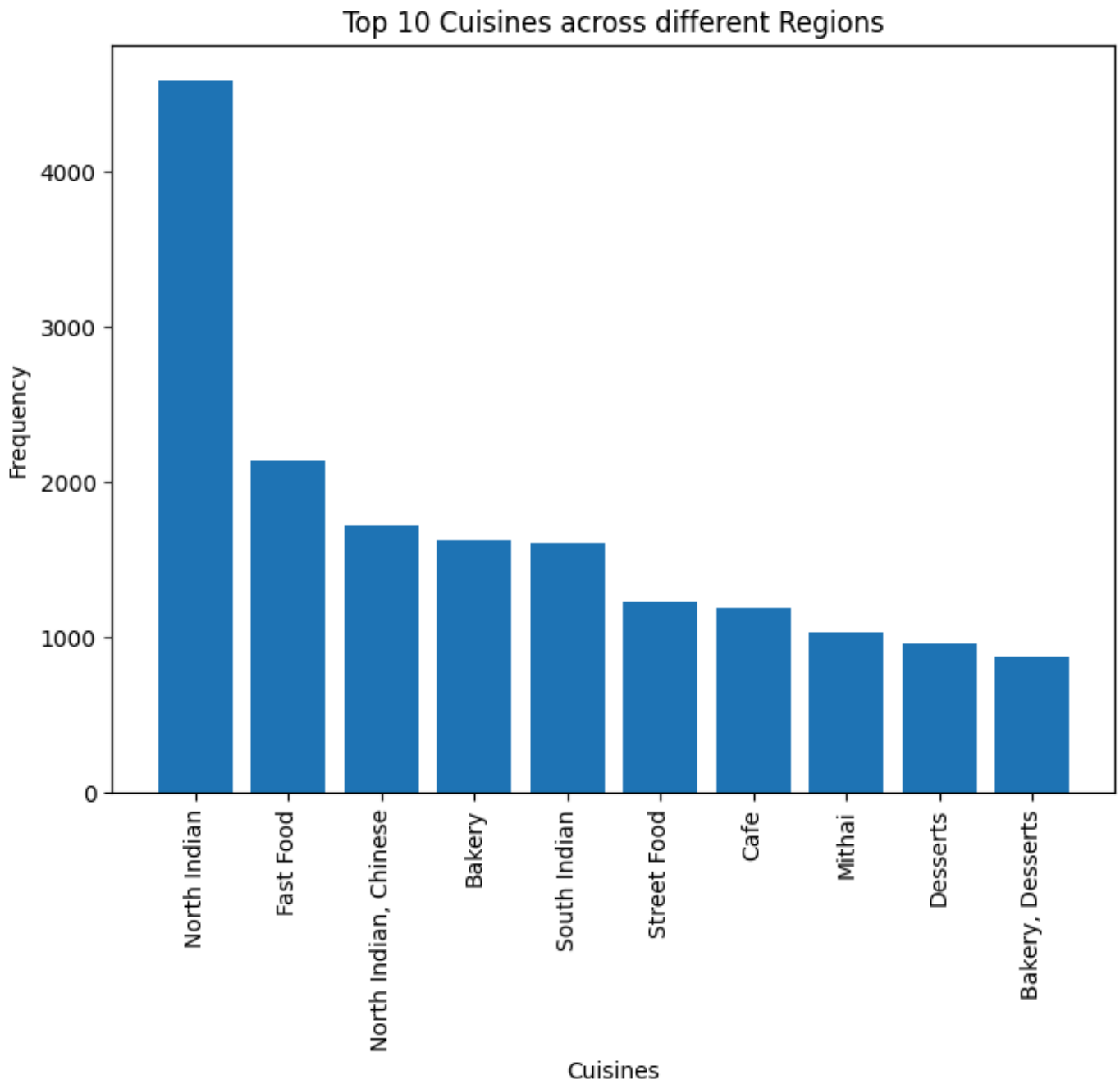
Out[52]:

	city	cuisines	aggregate_rating
21243	Zirakpur	Andhra, Goan, North Indian, Kerala	4.6
21108	Vizag	European, Mediterranean, North Indian	4.9
20903	Vijayawada	North Indian, Andhra	4.9
20697	Vellore	Juices, Italian, Burger	3.8
20529	Varanasi	North Indian, Chinese, BBQ	4.9
...	...	...	...
921	Allahabad	North Indian, Mediterranean	4.7
715	Alappuzha	Arabian, Continental	4.0
587	Ajmer	Continental, Beverages, South Indian, Fast Foo...	4.8
304	Ahmedabad	Chinese, Japanese	4.9
168	Agra	North Indian, Continental, Italian	4.9

99 rows × 3 columns

In [53]:

```
cuisines_region = df2.groupby('cuisines')['city'].size().sort_values(ascending=False)
top_cuisines = cuisines_region.head(10)
plt.figure(figsize=(8,6))
plt.bar(top_cuisines.index,top_cuisines.values)
plt.title('Top 10 Cuisines across different Regions')
plt.xlabel('Cuisines')
plt.ylabel('Frequency')
plt.xticks(rotation=90)
plt.show()
```

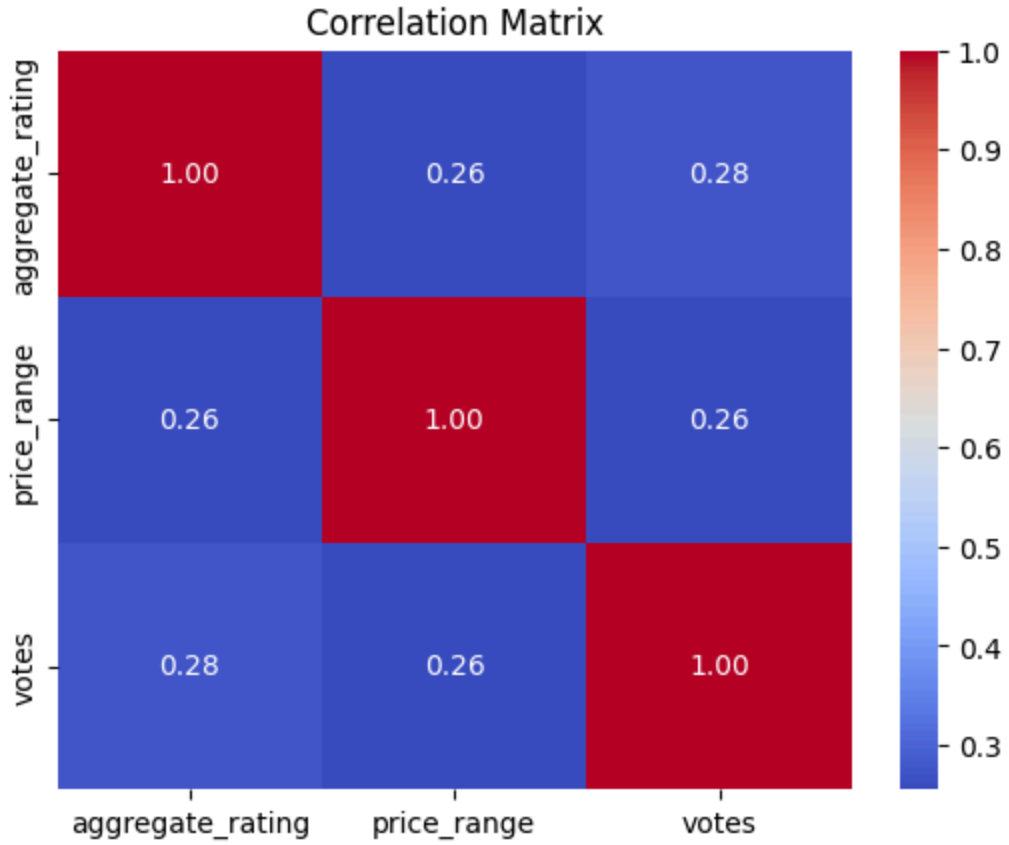


In [54]:

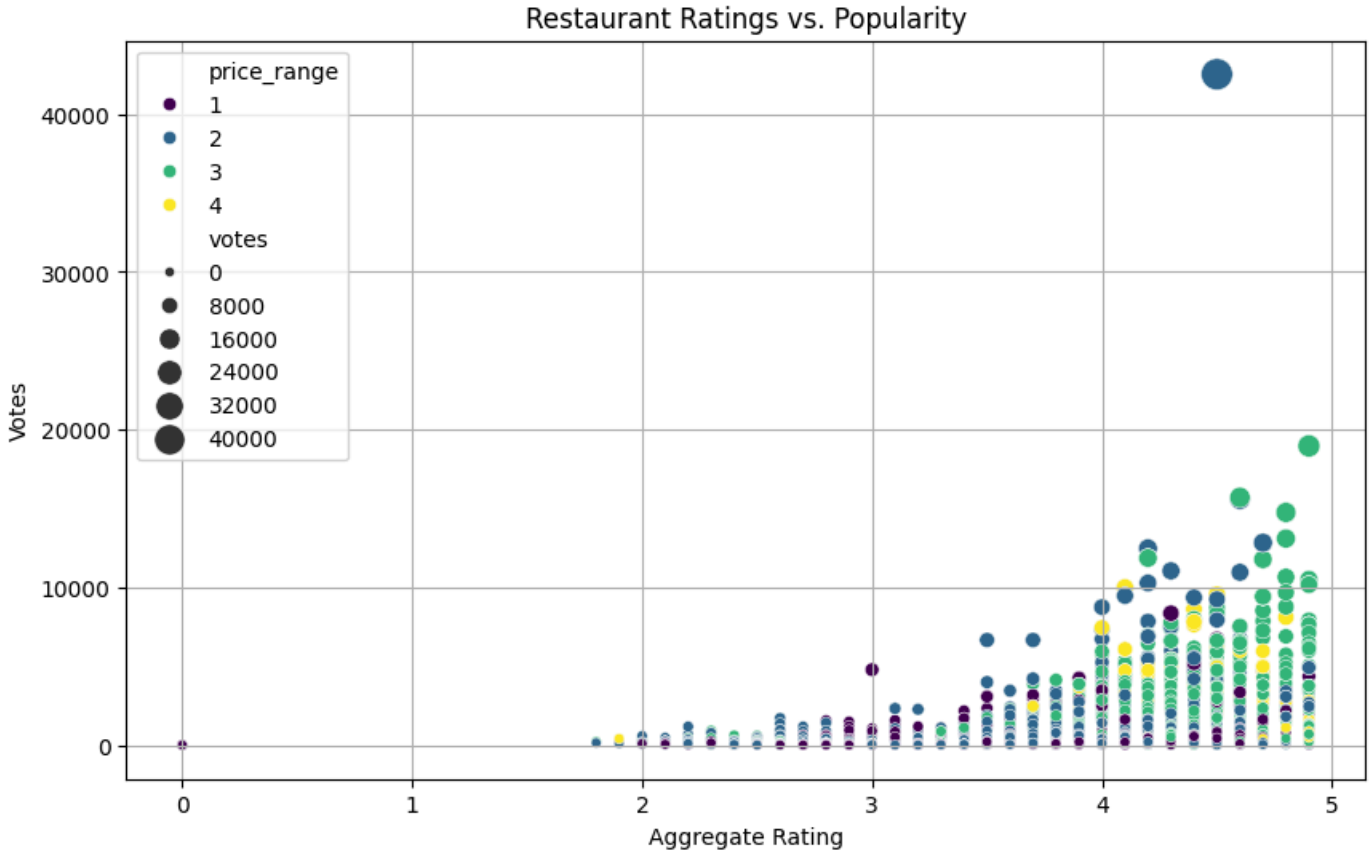
```
correlation_matrix = df2[['aggregate_rating', 'price_range', 'votes']].corr()
print("Correlation matrix:\n", correlation_matrix)

# Visualizing the correlation matrix
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()

# Scatter plot of ratings vs. price range
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df2, x='aggregate_rating', y='price_range', size='votes', alpha=0.6, sizes=
plt.title('Restaurant Ratings vs. Price Range')
plt.xlabel('Aggregate Rating')
plt.ylabel('Price Range')
plt.grid(True)
plt.show()

# Scatter plot of ratings vs. votes (popularity)
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df2, x='aggregate_rating', y='votes', hue='price_range', palette='viridis',
```

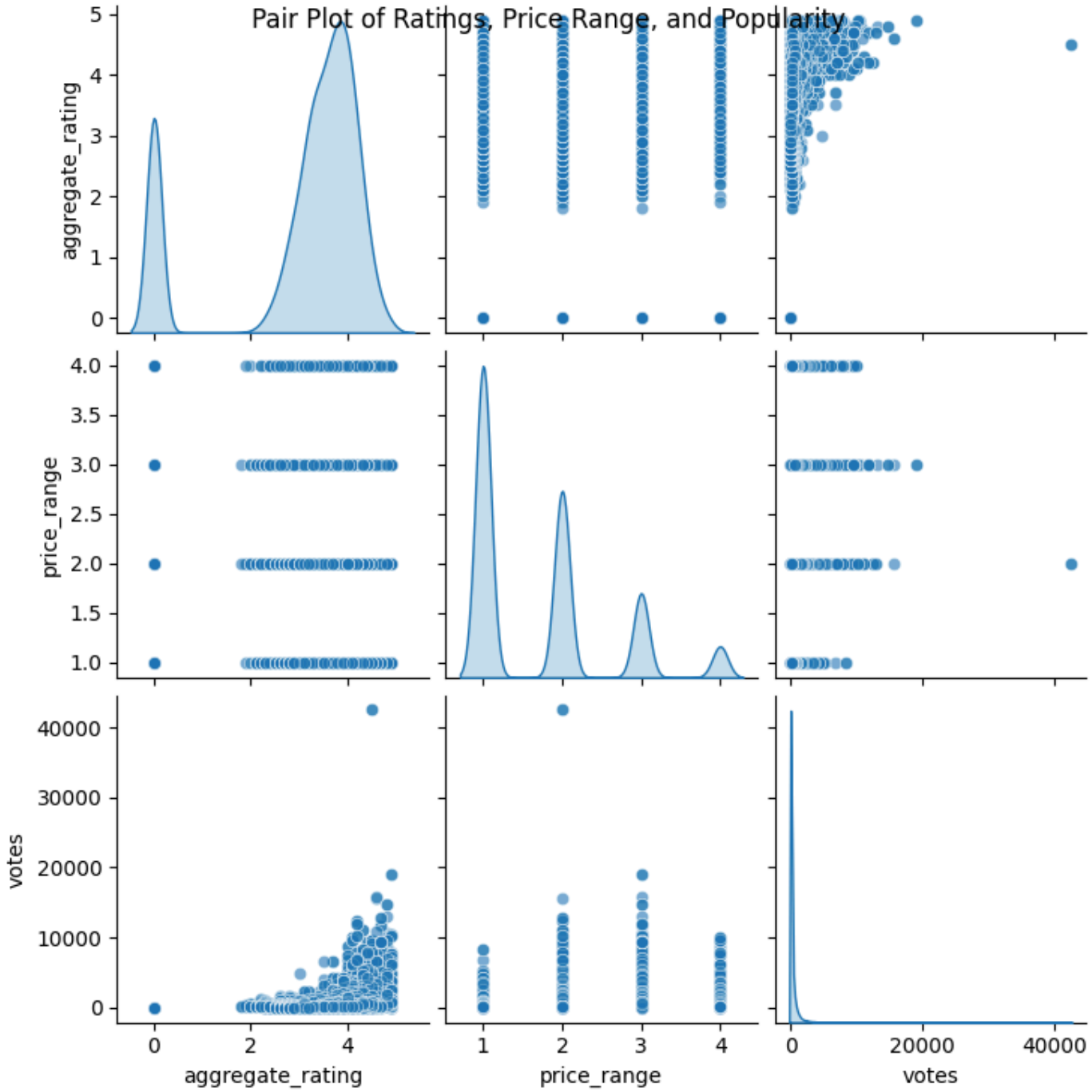
```
plt.title('Restaurant Ratings vs. Popularity')
plt.xlabel('Aggregate Rating')
plt.ylabel('Votes')
plt.grid(True)
plt.show()

# Optional: Pair Plot for a broader view
sns.pairplot(df2[['aggregate_rating', 'price_range', 'votes']], diag_kind='kde', plot_kws={'alph
plt.suptitle('Pair Plot of Ratings, Price Range, and Popularity', verticalalignment='top')
plt.show()
```

```
Correlation matrix:
                  aggregate_rating  price_range     votes
aggregate_rating          1.000000     0.257654  0.277529
price_range               0.257654     1.000000  0.255658
votes                     0.277529     0.255658  1.000000
```




Restaurant Ratings vs. Price Range
votes
0
8000
16000
24000
32000
40000
Price Range
Aggregate Rating
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0
1
2
3
4
5


Restaurant Ratings vs. Popularity
price_range
1
2
3
4
votes
0
8000
16000
24000
32000
40000
Votes
Aggregate Rating
40000
30000
20000
10000
0
0
1
2
3
4
5



Competitive Analysis

In [55]:

```
city_profile = df2.groupby('city').agg(
    highest_occuring_name=('name', lambda x: x.mode().iloc[0]),  # Get the mode (most frequent)
    most_number_of_establishments=('establishment', lambda x: x.mode().iloc[0]),  # Get the mode
    aggregate_rating=('aggregate_rating', 'mean'),  # Get the mean aggregate rating in the city
    top_cuisine=('cuisines', lambda x: x.mode().iloc[0]), # Get the mode cuisine in the city
    mostoccuringpricerange=('price_range', lambda x: x.mode().iloc[0]) #Get the mode for cuisine
).reset_index()

# Display the city profiles
print(city_profile)
```

```
          city highest_occuring_name most_number_of_establishments  \
0         Agra         Bhagat Halwai                ['Quick Bites']
1    Ahmedabad        Domino's Pizza             ['Casual Dining']
2        Ajmer   Gangaur Pizza Point                ['Quick Bites']
3    Alappuzha           Best Bakery                ['Quick Bites']
4    Allahabad        Baskin Robbins                ['Quick Bites']
..         ...                   ...                            ...
94    Varanasi             Chestnuts                ['Quick Bites']
95     Vellore        Domino's Pizza                ['Quick Bites']
96  Vijayawada           Sweet Magic                ['Quick Bites']
97      Vizag   Fresh Choice Bakery                ['Quick Bites']
98    Zirakpur        Baskin Robbins                ['Quick Bites']

    aggregate_rating   top_cuisine  mostoccuringpricerange
0           2.745404  North Indian                       1
1           3.266591   Street Food                       2
2           2.790426  North Indian                       1
3           0.525843        Bakery                       1
4           2.771252  North Indian                       1
..               ...           ...                     ...
94          3.037793  North Indian                       2
95          2.301466  South Indian                       1
96          3.066031  South Indian                       1
97          3.189182  South Indian                       1
98          2.827273  North Indian                       1

[99 rows x 6 columns]
```

Market Gap

In [56]:
```
df2[['delivery']]
```

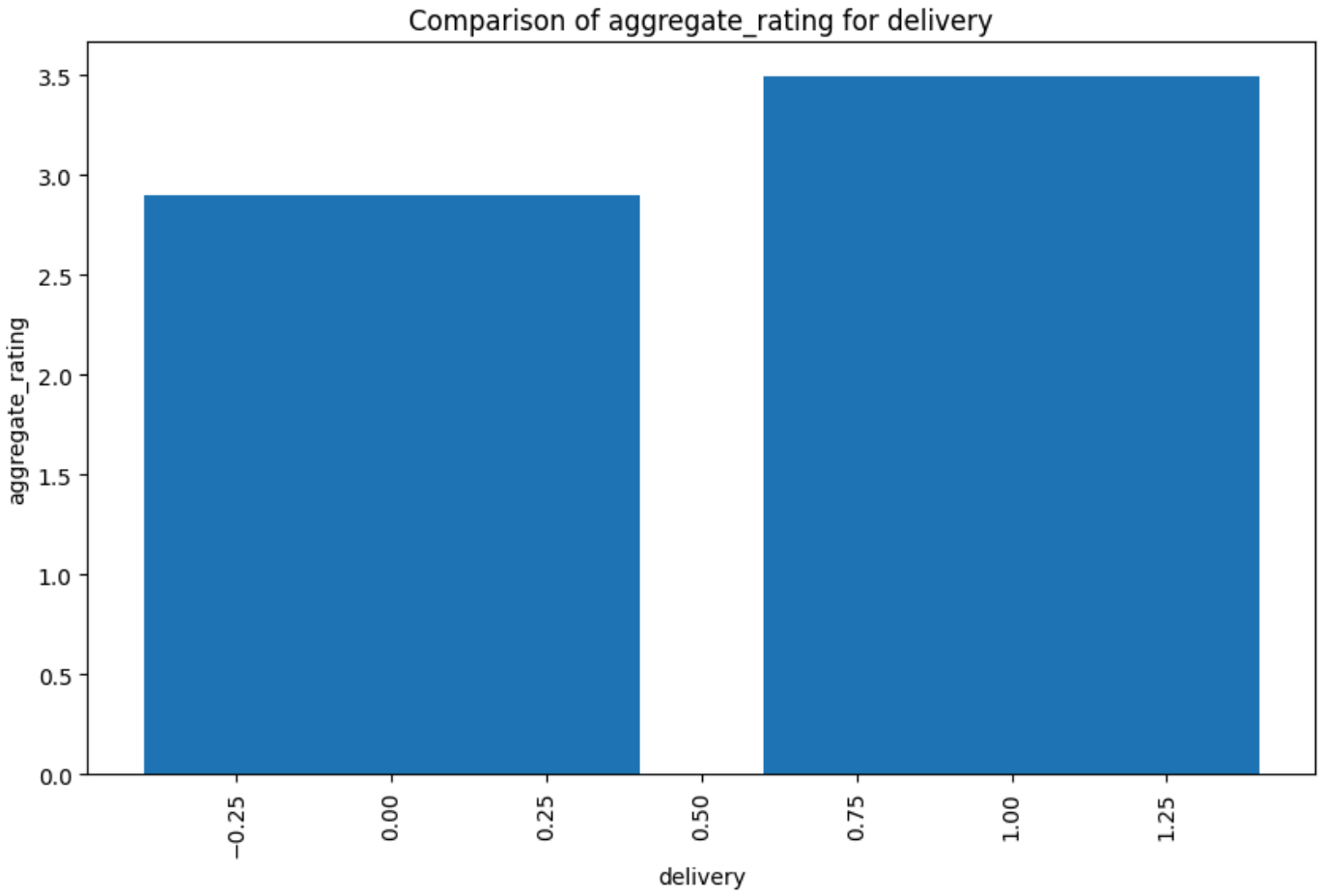
Out[56]:

	delivery
0	-1
1	-1
2	1
3	1
4	1
...	...
211882	1
211925	-1
211926	-1
211940	1
211942	1

60411 rows × 1 columns

In [57]:
```
delivery = df2.groupby('delivery')['aggregate_rating'].mean().reset_index()

mindev = 0
delivery = delivery[delivery['delivery'] >= mindev]
```

```
plt.figure(figsize=(10, 6))
plt.bar(delivery['delivery'], delivery['aggregate_rating'])
# Add labels and title
plt.xlabel('delivery')
plt.ylabel('aggregate_rating')
plt.title('Comparison of aggregate_rating for delivery')
plt.xticks(rotation=90)

# Show the plot
plt.show()
```

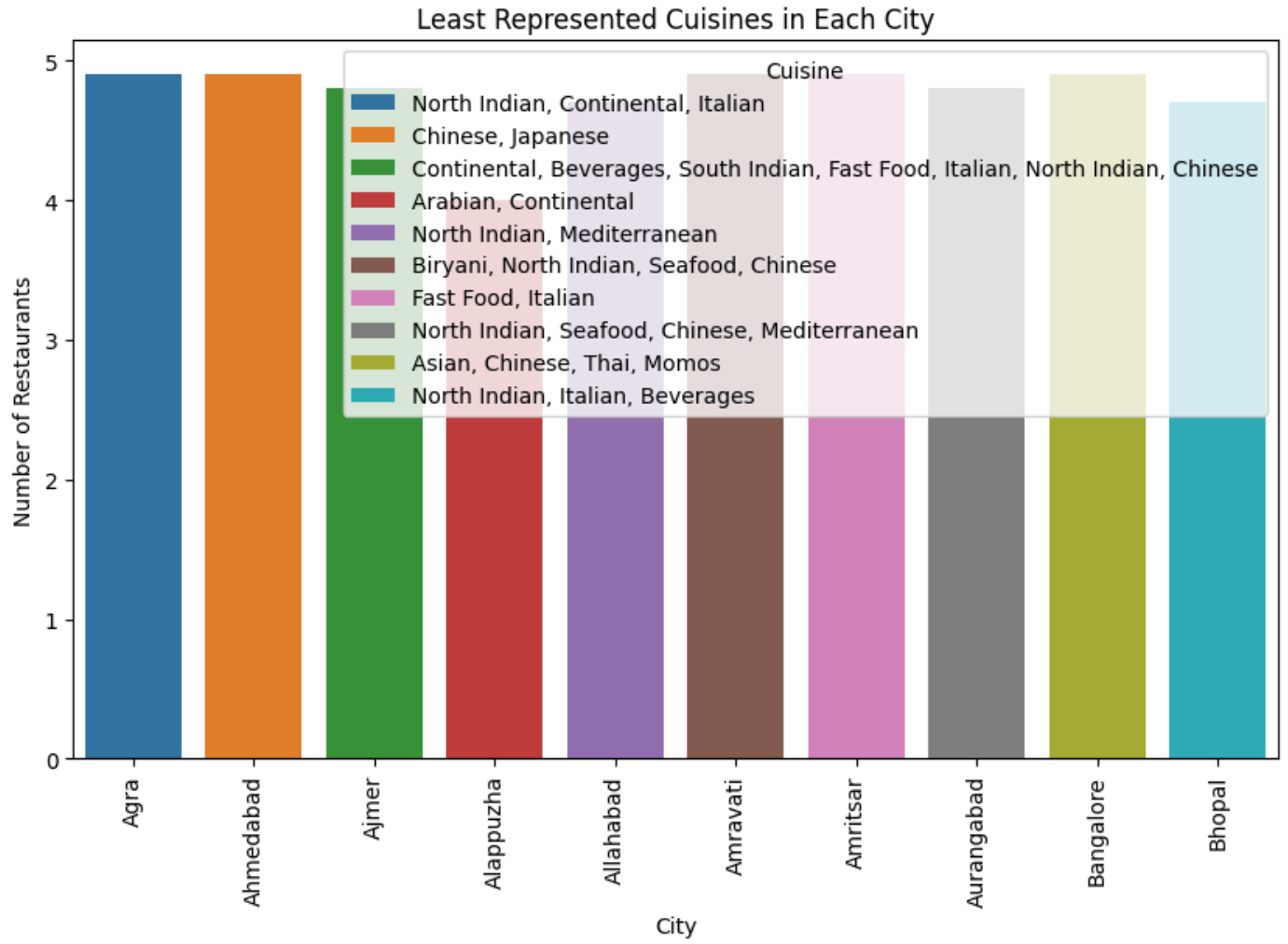


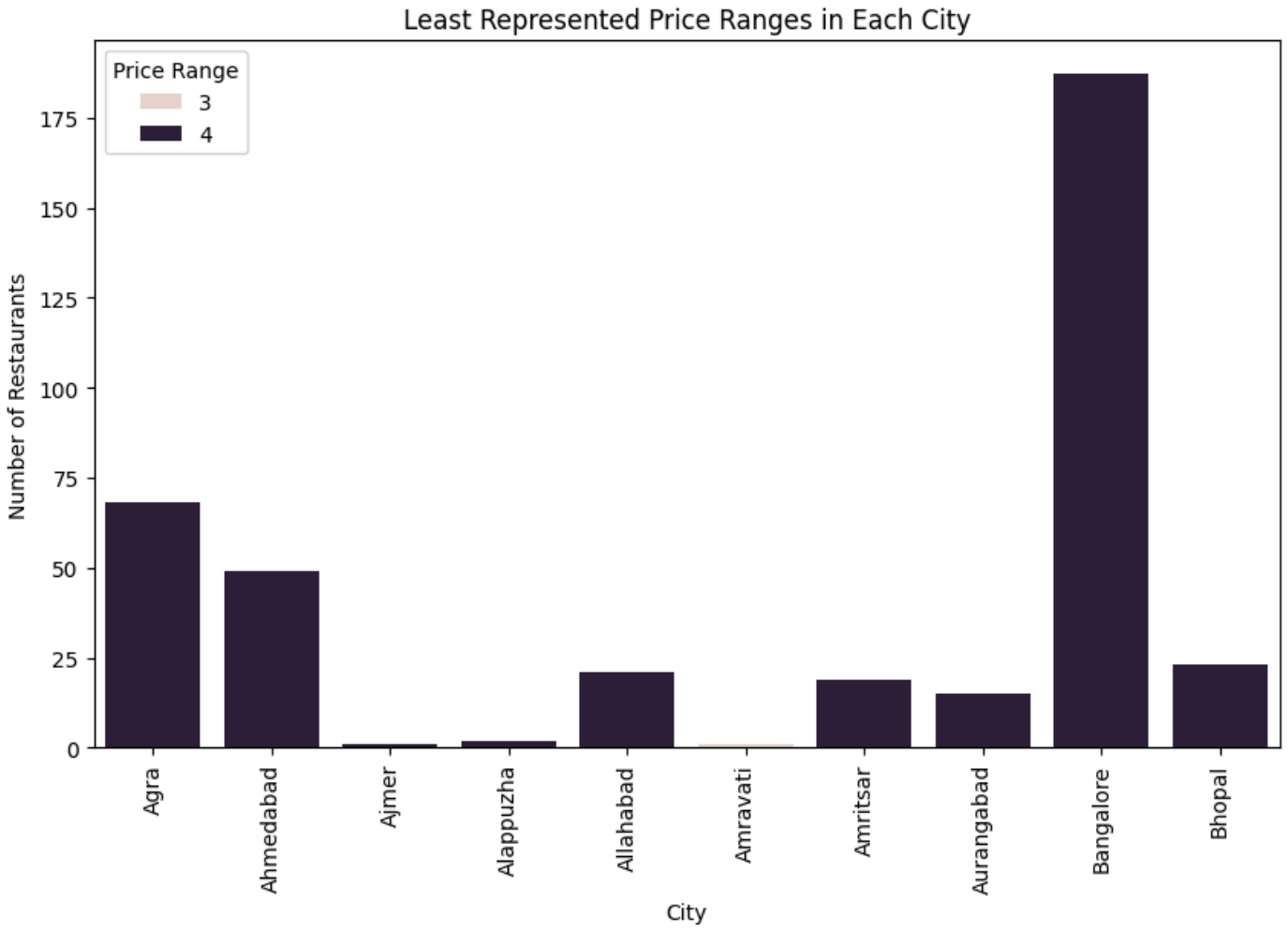
The gap in the market is for restaurant to determine that having delivery option raises the rating. Moreover, all places having rating more than 3 should be targeted by Zomato. For them even Zomato for delivery should be an option to cater to the requirements for the customers.

In [58]:
```
cuisine_distribution = df2.groupby(['city', 'cuisines'])['aggregate_rating'].mean().reset_index(
cuisine_distribution = cuisine_distribution.sort_values(by = ['city','aggregate_rating'], ascend
cuisine_distribution = cuisine_distribution.groupby('city').head(1)
price_range_distribution = df2.groupby(['city', 'price_range']).size().reset_index(name='count')
# Finding the least common cuisines in each city
least_common_cuisines = cuisine_distribution.groupby('city').apply(lambda x: x[x['aggregate_rati
least_common_cuisines = least_common_cuisines.head(10)
# Finding the least common price ranges in each city
least_common_price_ranges = price_range_distribution.groupby('city').apply(lambda x: x[x['count'
least_common_price_ranges = least_common_price_ranges.head(10)
```

In [59]:
```
# Plotting the least common cuisines in each city
plt.figure(figsize=(10, 6))
sns.barplot(data=least_common_cuisines, x='city', y='aggregate_rating', hue='cuisines')
plt.title('Least Represented Cuisines in Each City')
```

```
plt.xlabel('City')
plt.ylabel('Number of Restaurants')
plt.legend(title='Cuisine')
plt.xticks(rotation=90)
plt.show()

# Plotting the least common price ranges in each city
plt.figure(figsize=(10, 6))
sns.barplot(data=least_common_price_ranges, x='city', y='count', hue='price_range')
plt.title('Least Represented Price Ranges in Each City')
plt.xlabel('City')
plt.ylabel('Number of Restaurants')
plt.legend(title='Price Range')
plt.xticks(rotation=90)
plt.show()
```





In [60]:
```
cuisine_distribution = df2.groupby(['city', 'cuisines'])['aggregate_rating'].mean().reset_index(
cuisine_distribution = cuisine_distribution.sort_values(by = ['city','aggregate_rating'], ascend
cuisine_distribution = cuisine_distribution.groupby('city').head(1)
cuisine_distribution
```

Out[60]:

	city	cuisines	aggregate_rating
168	Agra	North Indian, Continental, Italian	4.9
304	Ahmedabad	Chinese, Japanese	4.9
587	Ajmer	Continental, Beverages, South Indian, Fast Foo...	4.8
715	Alappuzha	Arabian, Continental	4.0
921	Allahabad	North Indian, Mediterranean	4.7
...	...	...	...
20529	Varanasi	North Indian, Chinese, BBQ	4.9
20697	Vellore	Juices, Italian, Burger	3.8
20903	Vijayawada	North Indian, Andhra	4.9
21108	Vizag	European, Mediterranean, North Indian	4.9
21243	Zirakpur	Andhra, Goan, North Indian, Kerala	4.6

99 rows × 3 columns

In [63]:
```
price_range_distribution
```

Out[63]:

	city	price_range	count
0	Agra	1	481
1	Agra	2	237
2	Agra	3	106
3	Agra	4	68
4	Ahmedabad	1	521
...	...	...	...
375	Vizag	4	24
376	Zirakpur	1	86
377	Zirakpur	2	41
378	Zirakpur	3	20
379	Zirakpur	4	7

380 rows × 3 columns

The least represented cuisines by city help you understand that Zomato should do marketing for those specific cuisines in those cities.